

SceneDiffuser++: City-Scale Traffic Simulation via a Generative World Model

Supplementary Material

Shuhan Tan^{2*} John Lambert¹ Hong Jeon¹ Sakshum Kulshrestha¹ Yijing Bai¹
Jing Luo¹ Dragomir Anguelov¹ Mingxing Tan¹ Chiyu Max Jiang¹
¹Waymo LLC ²UT Austin

1. Supplementary Video

In our attached supplementary video, we include a brief and intuitive overview of:

- The city-scale traffic simulation task.
- SceneDiffuser++ architecture and training process.
- Rollout videos of SceneDiffuser++ across long horizons.

We encourage the readers to watch our supplementary video for a better understanding of the long simulation rollout quality from SceneDiffuser++.

2. Model Implementation Details

Architecture We use the same context encoder and Transformer denoiser backbone architecture as SceneDiffuser [2]. Our scene encoder architecture uses 192 latent queries. Each scene token is 512-dimensional, with 8 transformer layers and 8 transformer heads, with a transformer model dimension of 512. We train and run inference with all 128 agents.

Training To train SceneDiffuser++, we use the Adafactor optimizer [3], with EMA (exponential moving average). We decay using Adam, with $\beta_1 = 0.9$, $\text{decay}_{adam} = 0.9999$, weight decay of 0.01, and clip gradient norms to 1.0. We use a train batch size of 1024, and train for 1.2M steps. We select the most competitive model based on validation set performance, for which we perform a final evaluation using the test set. We use an initial learning rate of 3×10^{-4} . We use 32 diffusion sampling steps. When training, we mix the behavior prediction (BP) task with the scene generation task, with probability 0.5. The randomized control mask is applied to both tasks.

Feature Normalization To preprocess features, we use scaling constants of $\frac{1}{80}$ for features x, y, z , and compute mean μ and standard deviation σ of features l, w, h .

We preprocess each agent feature f to produce normalized feature f' via $f' = \frac{f - \mu_f}{2 * \sigma_f}$, where:

$$\mu_l = 4.5, \quad \mu_w = 2.0, \quad \mu_h = 1.75, \quad \mu_k = 0.5. \quad (1)$$

and

$$\sigma_l = 2.5, \quad \sigma_w = 0.8, \quad \sigma_h = 0.6, \quad \sigma_k = 0.5. \quad (2)$$

We scale by twice the std σ values to allow sufficient dynamic range for high feature values for some channels.

We conduct a similar feature normalization process for traffic light features. Specifically, we use the same scaling constants of $\frac{1}{80}$ for features x, y, z . We also convert the traffic light validity and one-shot state features to the range of $[-1, 1]$, similar to what we do for agent validity and type features.

3. Additional Results

In Figures 1 and 2, we show more qualitative results of SceneDiffuser++. Each row depicts a visualization of a 60-second trip-level rollout of our model. Within each row, we first show the full trip route overview (in the first column), and then subsequently plot SceneDiffuser++’s predictions at intervals corresponding to 0, 10, 15, 20 and 60 seconds from the start of simulation.

4. Experiment Details

Validity Definition We define a valid timestep for an agent as whether or not that agent appears in the AV’s detection output at a particular timestep. The Entering (or Exiting) Distance is the distance between an agent and the AV, in meters, at the timestep it appears in the AV’s detection for the first (or last) time.

Routing Implementation Details SceneDiffuser and SceneDiffuser++ do not use goal-oriented routing; in other words, they do not use or ingest a goal location in any way, shape, or form. Fig. 1 depicts with a star the “trip end” point, i.e. the final goal of the ADV, but there is no description of how the model is conditioned with the goal. This is because the main focus of this work is on the world model, while we assume that the planner can utilize any goal- or route-conditioned model for AV control. Therefore, in our experiments we also do not define a goal or progress-oriented metrics.

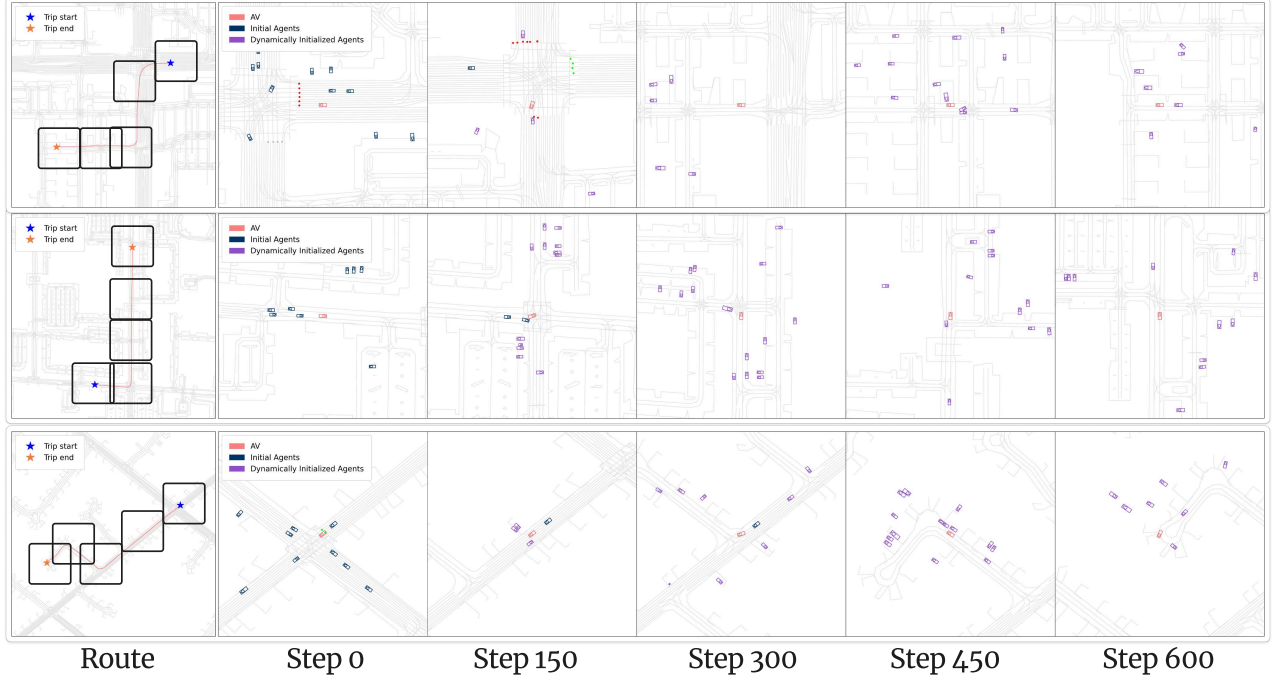


Figure 1. Additional qualitative results of SceneDiffuser++. (From left to right) Snapshots of full 60s rollouts at 0, 15, 30, 45 and 60-second timesteps.

When used as a planner, IDM explicitly searches for a valid path for the AV from the start location to a randomly selected goal location with a graph search algorithm on the WOMD lane graph. Similarly, when used as a world model, IDM searches for a path for every other agent.

For SceneDiffuser++, when used as a planner, we perform a route-unconditioned rollout in the mapped environment.

This is the same for all other agents when used as a world model. In this way, agents controlled by any of these three models follow a random path in the mapped environment, making it possible for us to only compare the realism aspect of the world models.

Traffic Light Transitions In order to quantitatively and qualitatively analyze the realism of simulated traffic light state transitions, we construct the traffic light transition probability matrix for SceneDiffuser++ and compare it against that of the ground-truth logs. We visualize the diagonals in Figure 8 of the main paper, and provide additional details below. Specifically, WOMD¹ [1] contains 9 different traffic light states: *Unknown*, *Green/Red/Yellow Arrow*, *Solid Green/Red/Yellow*, and *Flashing Red/Yellow*. Specifically, *Unknown* represents the case when the AV can observe the position of the traffic light, but cannot identify its state due to occlusion. We would like the model to predict only realistic traffic light state transitions, e.g. from *Yellow*

to *Red*, but not the other way around.

To compute the transition probability matrix, we count all the consecutive timesteps where the traffic light state changes from one state to a different state, and categorize them based on the starting state and end state, accumulating counts in a 9×9 transition matrix. As we observe that self-transitions from one state to itself are predominant, we removed all the self-transition counts (i.e., the diagonal entries on the transition matrix), and normalize the transition matrix to probabilities. We obtain the matrix in Figure 8 by computing an average over all scenarios in the validation dataset. To compute the JS-divergence between the transition probabilities for a quantitative comparison, we directly compute the JS-divergence between the ground-truth transition matrix and that produced by SceneDiffuser++. We observe that SceneDiffuser++ produces very realistic traffic light transitions.

References

- [1] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aurélien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021. 2
- [2] Chiyu Max Jiang, Yijing Bai, Andre Cornman, Christopher Davis, Xiukun Huang, Hong Jeon, Sakshum Kul-

¹<https://waymo.com/open/data/motion/tfexample/>

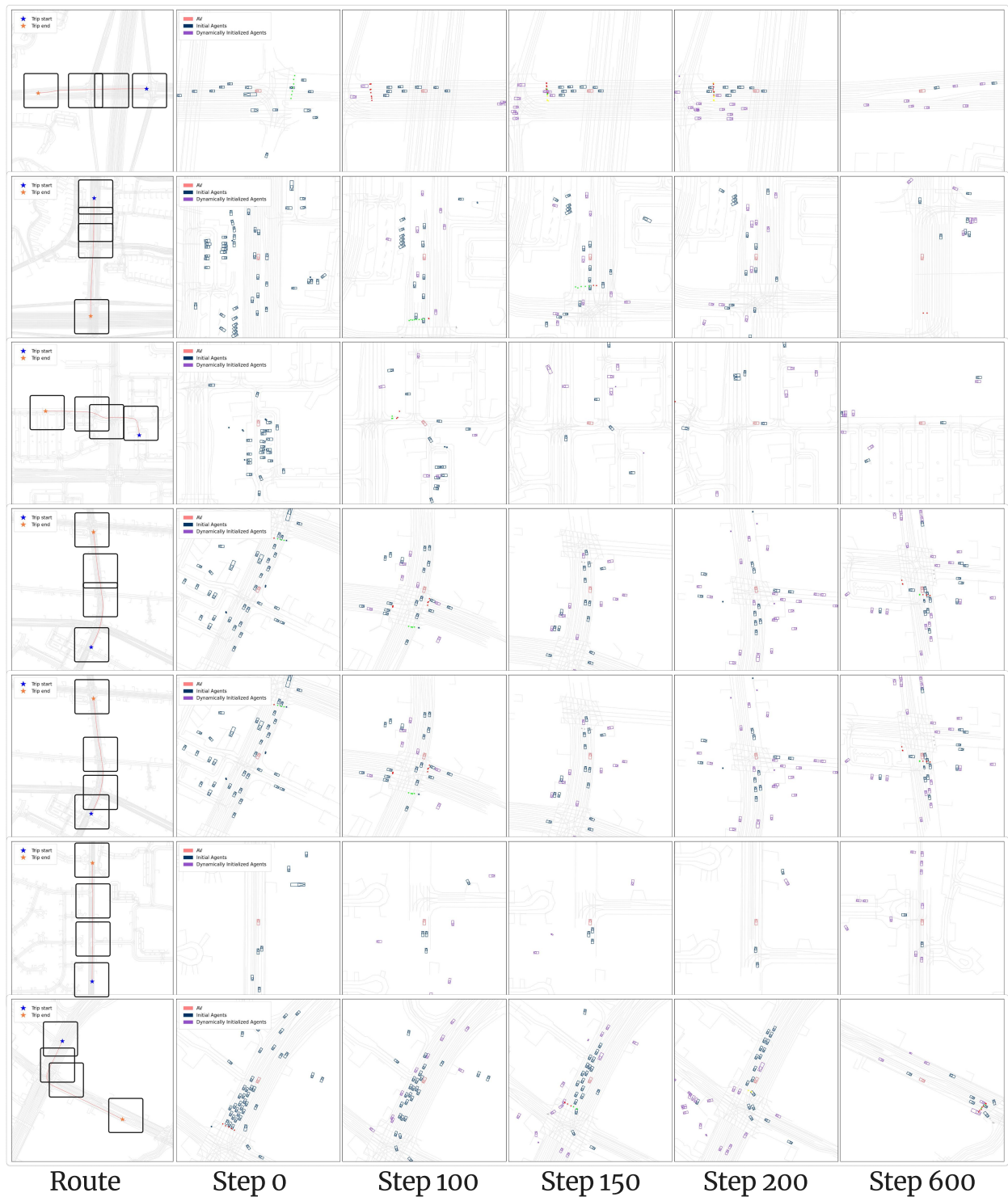


Figure 2. Additional qualitative results of SceneDiffuser++. (From left to right) Snapshots of full 60s rollouts at 0, 10, 15, 20 and 60-second timesteps.

shrestha, John Lambert, Shuangyu Li, Xuanyu Zhou, Carlos Fuertes, Chang Yuan, Mingxing Tan, Yin Zhou, and Dragomir Anguelov. Scenediffuser: Efficient and controllable driving simulation initialization and rollout. In *NeurIPS*, 2024. [1](#)

- [3] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *ICML*, 2018. [1](#)